IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re the Application of:		Atty. Docket No.:	003797.00626
Brumme et al.			
Serial No.:	10/633,911	Group Art Unit:	2193
Filed:	August 4, 2003	Examiner:	Chavis, John Q.
For:	Communication Among Agile and Context Bound Objects	Confirmation No.:	7569

BRIEF ON APPEAL

Mail Stop: Appeal Brief-Patents
Commissioner of Patents
P.O. Box 1450
Alexandria, VA 22313-1450

This is an appeal brief in accordance with 37 C.F.R. § 1.192 filed in support of applicant's July 27, 2007, Notice of Appeal. Appeal is taken from the final office action mailed January 29, 2007, and the advisory action mailed May 9, 2007. Please charge any necessary fees in connection with this appeal brief to our deposit account no. 19-0733.

I. REAL PARTY IN INTEREST

The owner of this application, and the real party in interest, is Microsoft Corporation.

II. RELATED APPEALS AND INTERFERENCES

There are no related appeals and interferences.

III. STATUS OF CLAIMS

Claims 1-17 remain in the application. All of the pending claims, claims 1-17, are shown in the attached appendix.

Claims 1-17 were rejected under 35 U.S.C. 102(e) as being anticipated by: (1) Kasichainula, et al., U.S. Patent No. 6,941,561; and (2) Montgomery, (NPL reference entitled Sunsoft's Object Lesson).

Applicants are appealing the rejections of claims 1-17. For the reasons set forth below, applicants respectfully submit that the final rejection of claims 1-17 is improper and should be reversed.

IV. STATUS OF AMENDMENTS

There have been no amendments filed in response to the final office action mailed January 29, 2007.

V. SUMMARY OF CLAIMED SUBJECT MATTER

In making reference herein to various portions of the specification and drawings in order to explain the claimed invention (as required by 37 C.F.R. § 41.37(c)(1)(v)), applicants do not intend to limit the claims; all references to the specification and drawings are illustrative unless otherwise explicitly stated.

Embodiments of the invention provide for efficient communication among agile objects and context bound objects within object-oriented programming environments, including communication between context-bound objects in different contexts, across contextual boundaries. (Page 2, line 22, through page 3, line 2).

Independent claim 1 is directed to a computer-implemented method for an object-oriented environment comprising: wrapping a reference to a second object within a second context with a proxy wrapper, the second context defining at least a second set of arbitrary invariants on a second set of arbitrary objects including the second object (page 10, line 1, through page 16, line 5; Figures 2 and 3); calling of the second object within the second context by a first object within a first context via the reference as wrapped in the proxy wrapper, the first context defining at least a first set of arbitrary invariants on a first set of arbitrary objects including the first object (page 10, line 1, through page 16, line 5; Figures 2 and 3); and, returning of the second object within the second context to the first object within the first context via the reference as wrapped in the proxy wrapper (page 13, line 11, through page 16, line 5, and Figure and 3).

Independent claim 6 is directed to A computerized system comprising: at least one first object within a first context, the first context defining at least a first set of arbitrary invariants on a first set of arbitrary objects including the at least one first object (page 10, lines 1-16 and Figure 2); and, at least one second object within a second context, the second context defining at least a second set of arbitrary invariants on a second set of arbitrary objects including the at least one second object (page 10, lines 1-16 and Figure 2), such that the at least one first object communicates with one another directly via direct references, the at least one second object communicates with one another directly via direct references, and any of the at least one first object communicates with any of the at least one second object via indirect references wrapped in proxy wrappers (page 10, line 1, through page 16, line 5; Figures 2 and 3).

Independent claim 12 is directed to a computerized system comprising: at least one first object within a first context, the first context defining at least a first set of arbitrary invariants on a first set of arbitrary objects including the at least one first object, the at least one first object communicating with one another directly via direct references (page 10, lines 1-16 and Figure 2); at least one second object within a second context, defining at least a second set of arbitrary invariants on a second set of arbitrary objects including the at least one second object, the at least one second object communicating with one another directly via direct references (page 10, lines 1-16 and Figure 2), and any of the at least one first object communicating with any of the at least one second object via indirect references wrapped in proxy wrappers (page 10, line 1, through page 16, line 5; Figures 2 and 3); and, at least one agile object, such that the at least one agile object are agile in that the at least one agile object have no permanent context, such as called by any of the at least one first object any of the at least one agile object executes within the first context, and such that as called by any of the at least one second object any of the at least one agile object executes within the second context (page 10, line 1, through page 16, line 5; Figures 2 and 3).

Independent claim 13 is directed to a machine-readable medium having a computer program stored thereon for execution by a processor to perform a method comprising: wrapping a reference to a second object within a second context with a proxy wrapper, the second context defining at least a second set of arbitrary invariants on a second set of arbitrary objects including the second object (page 10, line 1, through page 16, line 5; Figures 2 and 3); calling of the second object within the second context by a first object within a first context via the reference as wrapped in the proxy wrapper, the first context defining at least a first set of arbitrary invariants on a first set of arbitrary objects including the first object (page 10, line 1, through page 16, line 5; Figures 2 and 3); and, returning of the second object within the second context to the first object within the first context via the reference as wrapped in the proxy wrapper (page 10, line 1, through page 16, line 5; Figures 2 and 3).

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

1) Claims 1-17 were rejected under 35 U.S.C. 102(e) as being anticipated by Kasichainula, et al., U.S. Patent No. 6,941,561.

2) Claims 1-17 were rejected under 35 U.S.C. 102(e) as being anticipated by Montgomery, (NPL reference entitled Sunsoft's Object Lesson).

VII. ARGUMENT

A. Kasichainula (U.S. Patent 6,941,561) does not anticipate claims 1-5 and 13-17 because Kasichainula does not disclose "wrapping a reference to a second object within a second context with a proxy wrapper," and Kasichainula does not disclose "the first context defining at least a first set of arbitrary invariants on a first set of arbitrary objects including the first object."

Kasichainula discloses a method of distributing a program written using object orientated programming so that portions of the written program may be executed on more than one computer across a network. Two proxy objects are generated dynamically to enable communication over the intervening network by intercepting calls. The proxy objects allow method calls written for local invocation to be invoked over a network. The two proxies cooperate to hide the fact that the objects actually reside on different machines from the programmer, thereby sparing the programmer any need to be aware of the distributed nature of the system when writing his code.

Independent claims 1 and 13 include the claimed feature of "wrapping a reference to a second object within a second context with a proxy wrapper." The final Office Action mailed January 29, 2007, (hereinafter referred to as "the Office Action") states and Applicants agree that Kasichainula does not disclose "wrapping" a reference. However, the Office Action considers the claimed feature of "wrapping a reference to a second object with a second context with a proxy wrapper" to be inherently provided by Kasichainula. In particular, the Office Action states that in Kasichainula "code is not changed to enable access to remote objects and the programmer is unaware of the change that occurs to enable access objects as if they were local (i.e. indirectly)." (Office Action, page 4). Applicants respectfully disagree with the Office Action's position. Applicants respectfully submit that Kasichainula does not disclose the claimed feature of "wrapping . . . with a proxy wrapper."

In Kasichainula, <u>two proxy objects</u> are generated dynamically and used to enable communication over an intervening network by intercepting calls. In particular, Kasichainula's specification at col. 5 lines 5-11 states "When the call returns a data stream, Y" 406 and Y' 405 cooperate to communicate the data that the data stream represents across the network and present it to the caller as if it were local data." However, the use of two proxy objects does not disclose the claimed feature of "<u>wrapping a reference</u> to a second object within a second context with <u>a proxy wrapper</u>. (Emphasis added). Therefore, for at least this reason independent claims 1 and 13 are distinguishable over Kasichainula.

Independent claims 1 and 13 are allowable for at least an additional reason. Claims 1 and 13 include the claimed feature of "the first context defining at least a first set of <u>arbitrary</u>

invariants on a first set of arbitrary objects including the first object." The Office Action cites Kasichainula at col. 8 lines 26-31 for support of this claimed feature. Col. 8 lines 26-31 of Kasichainula states:

> [O]bject Y 503 may update a counter in its copy of object Z. Before the copy of object Z is returned to machine 501, another object in machine 501 may update the same counter in the actual object Z residing on machine 501. When object Y on machine 509 completes its operation on its copy of object Z, the object must be recopied to machine 501. . . .

Applicants respectfully submit that the cited portion of Kasichainula does not even remotely disclose defining a set of arbitrary invariants. In fact, the Office Action does not even mention the claimed "arbitrary invariants." Therefore, for at least this reason, Applicants respectfully submit that independent claims 1 and 13 are in condition for allowance.

Dependent claims 2-5 and 14-17 are in condition for allowance for at least the same reasons as independent claims 1 and 13 from which claims 2-5 and 14-17 ultimately depend.

B. Kasichainula (U.S. Patent 6,941,561) does not anticipate claims 6-11 because Kasichainula does not disclose "at least one first object communicates with any of the at least one second object via indirect references wrapped in proxy wrappers."

Independent claim 6 includes the claimed feature of "at least one first object communicates with any of the at least one second object via indirect references wrapped in proxy wrappers." (Emphasis added). Applicants respectfully submit that a proxy wrapper which includes indirect references may not properly be equated with the creation of two proxy objects in Kasichainula. Therefore, for at least this reason, Applicants respectfully submit that independent claim 6 is in condition for allowance.

Dependent claims 7-11 which ultimately depend from independent claim 6 are in condition for allowance for at least the same reason as independent claim 6.

C. Kasichainula (U.S. Patent 6,941,561) does not anticipate claim 12 because Kasichainula does not disclose "at least one agile object . . . in that the at least one agile object have no permanent context."

Independent claim 12 includes the claimed feature of "at least one agile object . . . in that the at least one agile object have no permanent context . . . ." With respect to independent claim 12, the Office Action states "the proxy objects are considered to provide the feature; since they are not fixed (generated dynamically via the abstract) to enable access as if they were local."

In an embodiment, Applicants' specification states on page 12-13:

> To a caller object within the context 200, the agile object 214 executes within the context 200 – that is, current execution thread for the agile object 214 is executes within the context 200 – such that to the caller object within the context 200 the agile object 214 appears as if it were context-bound in the context 200 for communication therewith. Likewise, to a caller object within the context 202, the agile object 214 executes within the context 202, such that to the caller object within the context 202 the agile object 214 appears as if it were context-bound in the context-bound in the context 202. The agile object 214 can directly access any other object in the context in which it is executing.

Applicants respectfully submit that an agile object is able to execute within multiple contexts. Kasichainula does not disclose objects that may execute within multiple contexts. Therefore, for at least this reason, Applicant respectfully submits that claim 12 is distinguishable over Kasichainula.

D. Montgomery (NPL reference entitled Sunsoft's Object Lesson) does not anticipate claims 1-5 and 13-17 because Montgomery does not disclose "wrapping a reference to a second object within a second context with a proxy wrapper."

Montgomery discloses a method the same source (and object) code is used to access objects within the same process, within the same machine but in different processes, within a network of cooperating machines.

Independent claims 1 and 13 include the claimed feature of "wrapping a reference to a second object within a second context with a proxy wrapper." The Office Action submits that Montgomery teaches a proxy wrapper by passing objects through a client stub. Montgomery states:

> stub marshals the argument – places them into a communication buffer with enough information so that the server can figure out what's going on. The stub then passes them to the subcontract which, in turn, executes the call to the server based object.

Applicants respectfully submit that marshalling an object into a communication buffer is not wrapping a reference to a second object within a second context with a proxy wrapper. Therefore, for at least this reason, Applicants respectfully submit that independent claims 1 and 13 are in condition for allowance.

Dependent claims 2-5 and 14-17 are in condition for allowance for at least the same reason as independent claims 1 and 13 from which they ultimately depend.

E. Montgomery (NPL reference entitled Sunsoft's Object Lesson) does not anticipate claim 12 because Montgomery does not disclose "at least one agile object. . . in that the at least one agile object have no permanent context."

Independent claim 12 includes the claimed feature of "at least one agile object. . . in that the at least one agile object have no permanent context . . . ." The Office Action submits that Montgomery discloses this claimed feature. The cited portion of Montgomery states:

> Spring's memory-management system is fairly simple. A client talks to an address space object. The address space maps to a memory object – e.g. a file. These two objects are the result of the cooperation of the virtual-memory manager (VMM) and an external pager.

The two objects mapping to each other in Montgomery do not disclose or suggest an agile object that has no permanent context. Therefore, for at least this reason, Applicants respectfully submit that independent claim 12 is distinguishable over Montgomery.

F. Montgomery (NPL reference entitled Sunsoft's Object Lesson) does not anticipate claims 6-11 because Montgomery does not disclose "at least one first object communicates with any of the at least one second object via indirect references wrapped in proxy wrappers."

Independent claim 6 includes the claimed feature of "at least one first object communicates with any of the at least one second object via <u>indirect references wrapped in proxy wrappers</u>." (Emphasis added). Applicants respectfully submit Montgomery does not disclose or suggest this claimed feature. Moreover, the Office Action does not specifically state where in Montgomery this claimed feature may be allegedly found. Therefore, for at least this reason, Applicants respectfully submits that independent claim 6 is in condition for allowance.

Dependent claims 7-11 which ultimately depend from independent claim 6 are in condition for allowance for at least the same reason as independent claim 6.

Respectfully submitted,
BANNER & WITCOFF, LTD.

Dated: November 27, 2007

By: /William J. Klein/
William J. Klein
Registration No. 43,719

10 S. Wacker Drive, Suite 3000
Chicago, IL 60606
(312) 463-5000

VIII. CLAIMS APPENDIX

1. A computer-implemented method for an object-oriented environment comprising:

wrapping a reference to a second object within a second context with a proxy wrapper, the second context defining at least a second set of arbitrary invariants on a second set of arbitrary objects including the second object;

calling of the second object within the second context by a first object within a first context via the reference as wrapped in the proxy wrapper, the first context defining at least a first set of arbitrary invariants on a first set of arbitrary objects including the first object; and,

returning of the second object within the second context to the first object within the first context via the reference as wrapped in the proxy wrapper.

2. The method of claim 1, further comprising:

wrapping a reference to the first object within the first context with a proxy wrapper, calling of the first object by the second object via the reference as wrapped in the proxy wrapper; and,

returning of the first object to the second object via the reference as wrapped in the proxy wrapper.

3. The method of claim 1, further comprising:

calling of a third object that is agile by the first object via an unwrapped direct reference, such that the first context of the first object becomes a context for the third object for calling of the third object by the first object, such that the third object executes in the first context of the first object, and such that the agile object is agile in that the agile object has no permanent context; and,

responding by the third object to the first object via the unwrapped direct reference.

4. The method of claim 3, further comprising:

calling of the first object by the third object via an unwrapped direct reference; and, responding by the first object to the third object via the unwrapped direct reference.

5. The method of claim 1, further comprising:

calling of a third object within the first context by the first object via an unwrapped direct reference; and,

responding by the third object to the first object via the unwrapped direct reference.

6. A computerized system comprising:

at least one first object within a first context, the first context defining at least a first set of arbitrary invariants on a first set of arbitrary objects including the at least one first object; and,

at least one second object within a second context, the second context defining at least a second set of arbitrary invariants on a second set of arbitrary objects including the at least one second object,

such that the at least one first object communicates with one another directly via direct references, the at least one second object communicates with one another directly via direct references, and any of the at least one first object communicates with any of the at least one second object via indirect references wrapped in proxy wrappers.

7. The system of claim 6, further comprising at least one agile object, such that the at least one agile object is agile in that the at least one agile object has no permanent context.

8. The system of claim 7, wherein as called by any of the at least one first object any of the at least one agile object executes within the first context, such that any of the at least first object communicates with any of the at least one agile object directly.

9. The system of claim 7, wherein as called by any of the at least one second object any of the at least one agile object executes within the second context, such that any of the at least second object communicates with any of the at least one agile object directly.

10. The system of claim 6, wherein a reference to one of the at least one second object is wrapped in a proxy wrapper, and one of the at least one first object calls the one of the at least one second object via the reference as wrapped in the proxy wrapper.

11. The system of claim 6, wherein a reference to one of the at least one first object is wrapped in a proxy wrapper, and one of the at least one second object calls the one of the at least one first object via the reference as wrapped in the proxy wrapper.

12. A computerized system comprising:

at least one first object within a first context, the first context defining at least a first set of arbitrary invariants on a first set of arbitrary objects including the at least one first object, the at least one first object communicating with one another directly via direct references;

at least one second object within a second context, defining at least a second set of arbitrary invariants on a second set of arbitrary objects including the at least one second object, the at least one second object communicating with one another directly via direct references, and any of the at least one first object communicating with any of the at least one second object via indirect references wrapped in proxy wrappers; and,

at least one agile object, such that the at least one agile object are agile in that the at least one agile object have no permanent context, such as called by any of the at least one first object any of the at least one agile object executes within the first context, and such that as called by any of the at least one second object any of the at least one agile object executes within the second context.

13. A machine-readable medium having a computer program stored thereon for execution by a processor to perform a method comprising:

wrapping a reference to a second object within a second context with a proxy wrapper, the second context defining at least a second set of arbitrary invariants on a second set of arbitrary objects including the second object;

calling of the second object within the second context by a first object within a first context via the reference as wrapped in the proxy wrapper, the first context defining at least a first set of arbitrary invariants on a first set of arbitrary objects including the first object; and, returning of the second object within the second context to the first object within the first context via the reference as wrapped in the proxy wrapper.

14. The medium of claim 13, further comprising:

wrapping a reference to the first object within the first context with a proxy wrapper;

calling of the first object by the second object via the reference as wrapped in the proxy wrapper; and,

returning of the first object to the second object via the reference as wrapped in the proxy wrapper.

15. The medium of claim 13, further comprising:

calling of a third object that is agile by the first object via an unwrapped direct reference, such that the first context of the first object becomes a context for the third object for calling of the third object by the first object, such that the third object executes in the first context of the first object, and such that the agile object is agile in that the agile object has no permanent context; and,

responding by the third object to the first object via the unwrapped direct reference.

16. The medium of claim 15, further comprising:

calling of the first object by the third object via an unwrapped direct reference; and,

responding by the first object to the third object via the unwrapped direct reference.

17. The medium of claim 13, further comprising:

calling of a third object within the first context by the first object via an unwrapped direct reference; and,

responding by the third object to the first object via the unwrapped direct reference.

IX. EVIDENCE APPENDIX

None.

X. RELATED PROCEEDINGS APPENDIX

None.